Safeguarding Hessian Approximations in Trust Region Algorithms

Richard G. Carter[1]

Technical Report 87-12, June 1987.

[1] Mathematical Sciences Department, Rice University, Houston, Texas, 77251-1892. Research sponsored by DOE DE-FG05-86ER25017, SDIO/IST/ARO DAAG-03-86-K-0113, and AFOSOR 85-0243.

Report Documentation Page	Form Approved OMB No. 0704-0188

1. REPORT DATE **JUN 1987**	2. REPORT TYPE	3. DATES COVERED **00-00-1987 to 00-00-1987**

4. TITLE AND SUBTITLE **Safeguarding Hessian Approximations in Trust Region Algorithms**	5a. CONTRACT NUMBER
	5b. GRANT NUMBER
	5c. PROGRAM ELEMENT NUMBER
6. AUTHOR(S)	5d. PROJECT NUMBER
	5e. TASK NUMBER
	5f. WORK UNIT NUMBER
7. PERFORMING ORGANIZATION NAME(S) AND ADDRESS(ES) **Computational and Applied Mathematics Department ,Rice University,6100 Main Street MS 134,Houston,TX,77005-1892**	8. PERFORMING ORGANIZATION REPORT NUMBER
9. SPONSORING/MONITORING AGENCY NAME(S) AND ADDRESS(ES)	10. SPONSOR/MONITOR'S ACRONYM(S)
	11. SPONSOR/MONITOR'S REPORT NUMBER(S)

12. DISTRIBUTION/AVAILABILITY STATEMENT
Approved for public release; distribution unlimited

13. SUPPLEMENTARY NOTES

14. ABSTRACT

15. SUBJECT TERMS

16. SECURITY CLASSIFICATION OF:			17. LIMITATION OF ABSTRACT	18. NUMBER OF PAGES	19a. NAME OF RESPONSIBLE PERSON
a. REPORT **unclassified**	b. ABSTRACT **unclassified**	c. THIS PAGE **unclassified**		**28**	

Standard Form 298 (Rev. 8-98)
Prescribed by ANSI Std Z39-18

Safeguarding Hessian Approximations in Trust Region Algorithms

Richard G. Carter

Abstract. In establishing global convergence results for trust region algorithms applied to unconstrained optimization, it is customary to assume either a uniform upper bound on the sequence of Hessian approximations or an upper bound linear in the iteration count. The former property has not been established for most commonly used secant updates, and the latter has only been established for some updates under the highly restrictive assumption of convexity.

One purpose of the uniform upper bound assumption is to establish a technical condition we refer to as the *uniform predicted decrease condition.* We show that this condition can also be obtained by milder assumptions, the simplest of which is a uniform upper bound on the sequence of Rayleigh quotients of the Hessian approximations in the gradient directions. This in turn suggests both a simple procedure for detecting questionable Hessian approximations and several natural procedures for *correcting* them when detected.

In numerical testing, one of these procedures increased the reliability of the popular BFGS method by a factor of two (i.e., the procedure halved the number of test cases to fail to converge to a critical point in a reasonable number of iterations). Further, for those problems where both methods were successful, this safeguarding procedure actually improved the average efficiency of the BFGS by ten to twenty percent.

Key words. Unconstrained optimization, trust region methods, secant methods, quasi-Newton methods, global convergence.

Acknowledgements. This research was sponsored by DOE DE-FG05-86ER25017, SDIO/IST/ARO DAAG-03-86-K-0113, and AFOSOR 85-0243. I would also like to thank John Dennis and Richard Tapia for their suggestions and support, and Vivian Choi for helping prepare the manuscript.

1. Introduction.

Trust region algorithms are an important class of iterative methods that can be used for solving the *unconstrained minimization problem*

$$\underset{x \in \mathbb{R}^n}{\text{minimize}}\, f(x) \tag{1.1}$$

where $f: \mathbb{R}^n \rightarrow \mathbb{R}$ is continuously differentiable with gradient $g: \mathbb{R}^n \rightarrow \mathbb{R}^n$ and Hessian $\nabla^2 f: \mathbb{R}^n \rightarrow \mathbb{R}^{n \times n}$. At each iteration k, the function f is approximated by the *quadratic model*

$$\psi_k(x) \equiv f(x_k) + g(x_k)^T(x - x_k) + \frac{1}{2}(x - x_k)^T B_k (x - x_k) \tag{1.2}$$

where $B_k \in \mathbb{R}^{n \times n}$ is a symmetric matrix intended to approximate $\nabla^2 f(x_k)$. A new iterate x_{k+1} is then generated by solving (perhaps approximately)

$$\begin{aligned} &\underset{x \in \mathbb{R}^n}{\text{minimize}}\, \psi_k(x) \\ &s/t \quad \|x - x_k\| \le \Delta_k \ . \end{aligned} \tag{1.3}$$

The positive scalar Δ_k, known as the *trust radius*, is adjusted at each iteration to ensure that ψ_k is a reasonably accurate approximation to f over the *trust region:* $\{x\colon \|x - x_k\| \le \Delta_k\}$. More complete descriptions can be found, for example, in [8], [13], [24], and [27].

The literature contains global convergence results for many different implementations of the trust region method. If the Hessian of f is uniformly bounded and f is bounded below, then the condition

$$\|B_k\| \le \beta_0, \quad 0 < \beta_0 < \infty \tag{1.4}$$

is sufficient to establish $\lim_{k \rightarrow \infty} \|g_k\| = 0$ for most implementations, while the condition

$$\|B_k\| \le \beta_0(1 + k), \quad 0 < \beta_0 < \infty \tag{1.5}$$

is sufficient to establish the weaker result $\liminf_{k \rightarrow \infty} \|g_k\| = 0$ (see, for example [24] and [22]).

In the event that an analytic expression for the Hessian is impractical and a finite difference approximation is too expensive, it is customary to use one of the *secant methods* to compute the model Hessian B_k ([7], [8]). Unfortunately, the question of whether these methods satisfy bounds of the form (1.4) remains open. Condition (1.5) can be established for the popular BFGS secant method, but only under the highly restrictive assumption that f is convex [20].

One of the ways that conditions (1.4) and (1.5) are used in proving global convergence is to establish that, for every iteration where $g(x_k) \neq 0$, the approximate solutions to (1.3) satisfy

$$\psi_k(x_k) - \psi_k(x_{k+1}) \geq \frac{1}{2}\beta_1 \, \| g(x_k) \| \min \left\{ \Delta_k, \frac{\| g(x_k) \|}{\beta_0} \right\} \tag{1.6}$$

or

$$\psi_k(x_k) - \psi_k(x_{k+1}) \geq \frac{1}{2}\beta_1 \, \| g(x_k) \| \min \left\{ \Delta_k, \frac{\| g(x_k) \|}{\beta_0 (1+k)} \right\} \tag{1.7}$$

respectively, where $\beta_1 > 0$. However, (1.6) and (1.7) can also be established by the milder conditions

$$g(x_k)^T B_k g(x_k) \leq \beta_0 \, g(x_k)^T g(x_k) \tag{1.8}$$

and

$$g(x_k)^T B_k g(x_k) \leq \beta_0 (1+k) \, g(x_k)^T g(x_k) \tag{1.9}$$

respectively.

Conditions (1.8) and (1.9) are no less of an open question for secant methods than are (1.4) and (1.5). However, several natural methods are available for *correcting* B_k so that $g(x_k)^T B_k g(x_k) \approx g(x_k)^T \nabla^2 f(x_k) g(x_k)$. Under the mild assumption that $\| \nabla^2 f(x_k) \|$ is uniformly bounded, these methods guarantee (1.8) and hence (1.6).

In this paper we present several procedures for safeguarding Hessian approximations. These procedures were designed with the following goals in mind.

(a) Condition (1.6) must be satisfied at every iteration for some constants β_0 and β_1 independent of k.

(b) The test used to determine when a correction is to be made should be inexpensive and should be as scale invariant as possible.

(c) Any corrections made should not change the invariance properties of the original method. Furthermore, if used in conjunction with a secant method which produces positive definite Hessian approximations, the corrections should also be positive definite.

(d) In numerical testing, the safeguards must improve the overall *reliability* of the trust region algorithm without significantly harming the *efficiency*.

Surprisingly, design goal (d) was actually exceeded by our procedures. Even though each correction involved extra work, the number of iterations typically needed for convergence decreased enough so that the overall efficiency was improved by 10 to 20% for our test set. Moreover, one of our procedures decreased the number of test cases that failed to converge in a reasonable number of iterations by a factor of two.

It should be pointed out that (1.8) is not *of itself* sufficient to imply global convergence. Bounds such as (1.4) and (1.5) are not only used to establish (1.6) or (1.7), but are also used to establish conditions about how the trust radius is updated at each iteration. However, it is shown in [2] that the sufficiency conditions on the trust radius updating procedure can be obtained if (1.4) is replaced by the weaker condition

$$-\beta_0 p^T p \le p^T B_k p \quad \forall \quad \textit{nonzero} \quad p \in \mathbb{R}^n \ . \tag{1.10}$$

That is, global convergence *can* be established provided (1.8) holds and the eigenvalues of B_k are bounded away from $-\infty$. For typical secant methods that force each B_k to be positive definite, the correction procedures presented in later sections are sufficient to give $\lim_{k \to \infty} \|g_k\| = 0$.

The remainder of this paper is organized as follows. Section 2 presents nomenclature and preliminary theory. Section 3 presents the procedures for safeguarding and proves that they enforce (1.6). Section 4 presents our test results, and describes some of the alternatives investigated during this study. We conclude the paper with a summary of results and a brief discussion of some possible refinements to our safeguarding techniques.

2. Preliminaries. The following notation is used throughout this paper. The function $f:\mathbb{R}^n \rightarrow \mathbb{R}^1$ has gradient $g:\mathbb{R}^n \rightarrow \mathbb{R}^n$ and Hessian $\nabla^2 f:\mathbb{R}^n \rightarrow \mathbb{R}^{n\times n}$. The vectors $\{x_k\}$ are the iterates produced by the trust region algorithm, and $p_k \equiv x_{k+1} - x_k$. For brevity, f_k will denote $f(x_k)$, g_k will denote $g(x_k)$, etc. The notation $\|\cdot\|$ denotes the Euclidean norm when applied to a vector and the operator norm induced by the Euclidean norm when applied to a matrix.

The *predicted function reduction* at the k^{th} iteration is defined by

$$\begin{aligned} pred_k(p) &\equiv f_k - \psi_k(x_k + p) \\ &= -g_k^T p - \frac{1}{2} p^T B_k p \ . \end{aligned} \tag{2.1}$$

The function $U(\cdots)$ represents whatever procedure we are originally using to generate or update our Hessian approximations $\{B_k\}$ (i.e., $B_{k+1} = U(B_k, p_k, g_{k+1} - g_k)$). The notation $\hat{w}$ denotes $w/\|w\|$ for nonzero $w \in \mathbb{R}^n$. For any nonzero $w \in \mathbb{R}^n$ and symmetric $B \in \mathbb{R}^{n\times n}$, $c(B,w)$ represents the *Rayleigh quotient* of B in the direction w:

$$c(B,w) = w^T B w / w^T w \ . \tag{2.2}$$

This notation is appropriate since $c(B_k, w)$ also represents the *curvature* of the quadratic model ψ_k in the direction w.

Using this notation, (1.6) becomes

$$pred_k(p_k) \geq \frac{1}{2}\beta_1 \|g_k\| \min\{\Delta_k, \|g_k\|/\beta_0\} \ . \tag{2.3}$$

We refer to this as the ***uniform predicted decrease condition.***[1] Similarly, (1.3) becomes

$$\begin{aligned} &minimize \ (-pred_k(p)) \\ &s/t \quad \|p\| \leq \Delta_k \ , \end{aligned} \tag{2.4}$$

or the *trust region subproblem*, and (1.8) becomes

$$c(B_k, g_k) \leq \beta_0 \ . \tag{2.5}$$

[1] This terminology helps distinguish between conditions (1.6) and (1.7). The word "uniform" refers to the fact that the denominator of the last term, β_0, is not a function of k.

A variety of methods are available for computing approximate solutions to the trust region subproblem. In general, these methods satisfy

$$pred_k(p_k) \geq \beta_2 \min \{-pred_k(p)\ s/t\ p \in S_k,\ \|p\| \leq \Delta_k\} \tag{2.6a}$$

and

$$\|p_k\| \leq \beta_3 \Delta_k\ , \tag{2.6b}$$

for some subspace S_k and constants $\beta_2 \in (0,1)$, $\beta_3 \geq 1$. Algorithms which satisfy (2.6) for the choice $S_k = \mathbb{R}^n$ are usually referred to as *Hebden-Moré* (see [12], [13], and [16]), *optimal locally constrained* (OLC) (see [10]), or *hookstep* (see [8]) methods, and typically require between one and two matrix factorizations per iteration. Other methods select a subspace S_k of smaller dimension in order to avoid these factorizations. The *dogleg* method of Powell [18] and the *double dogleg* of Dennis and Mei [6] solve the trust region subproblem over a piecewise linear path embedded in $span\ \{-g_k, -B_k^{-1}g_k\}$. Steihaug [28] considers a conjugate gradient method in which a sequence of trial steps p_k^i, $i = 1, 2, \cdots$ are generated that minimize the quadratic model over a sequence of expanding Krylov subspaces $S_k^i = span\ \{-g_k, B, -B_k g_k, ..., -B_k{}^i g_k\}$. The procedure continues until either p_k is a sufficiently accurate global minimizer of the quadratic, or the piecewise linear path defined by the trial steps intersects the boundary of the trust region. All of these methods satisfy (2.6) with S_k defined to be $span\ \{-g_k\}$. Given this minimal property, it is fairly straightforward to show that (2.5) implies the uniform predicted decrease condition (2.3).

LEMMA 1. Let w_k and g_k be nonzero vectors in $\mathbb{R}^n$ and let Θ_k be the angle between w_k and $-g_k$. Consider the problem

$$\begin{aligned} &minimize\ \ (-pred_k(p)) \\ &s/t\ \ \ p = \alpha w_k\ ,\ \|p\| \leq \Delta_k \end{aligned} \tag{2.7}$$

where $c(B_k, w_k)$ and $\cos\Theta_k$ are not both zero. A vector p^* solves (2.7) only if

$$p^* = \frac{\|g_k\| \cos\Theta_k}{c(B_k, w_k)}\ \hat{w}_k \tag{2.8}$$

with $\|p^*\| \leq \Delta_k$ and $c(B_k, w_k) > 0$, or

$$p^* = \Delta_k \hat{w}_k \,, \tag{2.9a}$$

or

$$p^* = -\Delta_k \hat{w}_k \,. \tag{2.9b}$$

Furthermore, if (2.8) solves (2.7) we have

$$pred_k(p^*) = \frac{1}{2} \, ||g_k||^2 \cos^2\Theta_k / c(B_k, w_k) \,, \tag{2.10}$$

while if (2.9a) or (2.9b) solves (2.7) we have

$$pred_k(p^*) \geq \frac{1}{2} \cos\Theta_k \; ||g_k|| \, \Delta_k \,. \tag{2.11}$$

Proof. Problem (2.7) is simply the minimization of a quadratic in one dimension subject to upper and lower bounds. Substituting $p = \alpha w_k$ and $\cos(\Theta_k) = -\, g_k^T w_k / (\, ||g_k|| \; ||w_k||)$ into (2.1), we can write

$$\begin{aligned} q(\alpha) &\equiv -pred(\,\alpha\, w_k\,) \\ &= -\,\alpha(\cos\Theta_k \; ||g_k|| \; ||w_k||) + \frac{\alpha^2}{2} \; ||w_k||^2 c(B_k, w_k) \end{aligned} \tag{2.12}$$

and transform problem (2.7) into

$$\begin{aligned} &\mathit{minimize} \; q(\alpha) \\ &s/t \;\; |\alpha| \leq \Delta_k / \, ||w_k|| \,. \end{aligned} \tag{2.13}$$

(i) If $c(B_k, w_k) > 0$, it is easy to establish that (2.8) and (2.10) hold provided the constraint is inactive. If the constraint is active, then (2.9) and (2.11) follow from (2.8) and (2.13) since $sign\ (\alpha^*) = sign\ (-\, g_k^T w_k)$ and $\alpha^* \; ||w_k|| \leq \cos\Theta_k \; ||g_k|| / c(B_k, w_k)$ so that

$$\begin{aligned} q(\,\alpha^*) &= -\,\alpha^* \; ||w_k|| \cos\Theta_k \; ||g_k|| \left\{ 1 - \frac{\alpha^*}{2} \; ||w_k|| \; \frac{c(B_k, w_k)}{\cos\Theta_k \; ||g_k||} \right\} \\ &\leq -\,\alpha^* \; ||w_k|| \cos\Theta_k \; ||g_k|| \{1 - ½\} \\ &\leq -\frac{1}{2} \Delta_k \cos\Theta_k \; ||g_k|| \,. \end{aligned} \tag{2.14}$$

Hence (2.11) holds.

(ii) If $c(B_k, w_k) \leq 0$, then the constraint is necessarily binding, and (2.7) is solved by (2.9a) and/or (2.9b). Then (2.12) immediately implies (2.11). □

THEOREM 2. If there exist $\beta_2 > 0$ and $\beta_3 > 1$ such that (2.6) is satisfied at every iteration and $g_k \in S_k$, then (2.5) is sufficient to imply the uniform predicted decrease condition with $\beta_1 = \beta_2$. Moreover, if $w_k \in S_k$ is a nonzero vector satisfying

$$c(B_k\,, w_k) \leq \beta_4 \tag{2.15}$$

for some $\beta_4 > 0$, then (2.6) implies

$$pred_k(p_k) \geq \frac{1}{2}\beta_2 \cos\Theta_k \,\|g_k\| \min\left\{\Delta_k, \cos\Theta_k \,\frac{\|g_k\|}{\beta_4}\right\}, \tag{2.16}$$

where Θ_k is the angle between w_k and $-g_k$.

Proof. Applying Lemma 1 to (2.6) with $w_k = -g_k$ immediately establishes that (2.5) implies (2.3). Equation (2.16) also follows immediately from Lemma 1. □

3. Safeguarding procedures. We now present three safeguarding procedures. The first is simple, direct, and can be applied to any method for generating Hessian approximations. This procedure first compares the curvature of the model in the gradient direction with an estimate of the maximum curvature of the problem. If the model curvature seems too large, the model Hessian is scaled using a finite difference approximation of the exact curvature $c(\nabla^2 f(x_{k+1}), g_{k+1})$. The positive constants *macheps* and *typx* represent machine epsilon and an estimate of the typical size of $\|x_k\|$.

Procedure 1

Given $k \geq 1$, $c_{k-1} > 0$, x_k, x_{k+1}, p_k, g_k, g_{k+1}, and B_k, do the following.

(1) Approximate $\nabla^2 f(x_{k+1})$ by some method: $B_{k+1} = U(B_k, \ldots)$.

(2) Estimate the maximum curvature of the problem:

$$c_k = \max\{c_{k-1}, p_k^T(g_{k+1} - g_k)/p_k^T p_k\} . \tag{3.1}$$

(3) If $c(B_{k+1}, g_{k+1}) > c_k$ then apply a correction to B_{k+1}:

(a) Approximate $c(\nabla^2 f(x_{k+1}), g_{k+1})$ by finite differences:

$$\epsilon = (macheps)^{\frac{1}{3}}(typx)/\,||\,g_{k+1}\,||\,, \tag{3.2a}$$

$$\bar{c}_k = 2\,\frac{f_{k+1} - f(x_{k+1} - \epsilon\, g_{k+1}) - \epsilon\, g_{k+1}^T g_{k+1}}{\epsilon^2 g_{k+1}^T g_{k+1}} . \tag{3.2b}$$

(b) Scale B_{k+1}:

If $\bar{c}_k > 0$, then set $\alpha = \bar{c}_k / c(B_{k+1}, g_{k+1})$,

else set $\alpha = c_k / c(B_{k+1}, g_{k+1})$.

Set $B_{k+1} := \alpha B_{k+1}$.

(4) Exit.

The next procedure is more sophisticated and makes use of the properties of secant updates explicitly. Such methods are designed to satisfy the equation

$$U(B, p, y)\,p = y . \tag{3.3}$$

Thus, if $p = p_k$ and $y = y_k \equiv g_{k+1} - g_k$, the update $B_{k+1} = U(B_k, p_k, y_k)$ will exactly interpolate the change in the gradient:

$$B_{k+1}(x_{k+1} - x_k) = g_{k+1} - g_k . \tag{3.4}$$

Such methods have proven to be very successful. Of particular importance are methods which have the property of *hereditary positive definiteness:*

$$(B_k \textit{ positive definite }) \Rightarrow (B_{k+1} \textit{ positive definite.}) \tag{3.5}$$

Many secant methods have this property provided $y_k^T p_k > 0$, so it is customary to forego the update if $y_k^T p_k \leq 0$. That is,

$$(\; y_k^T p_k \leq 0 \;) \Rightarrow (\; B_{k+1} = B_k \;). \tag{3.6}$$

Updates that satisfy (3.5) and (3.6) are said to be ***positive definite secant updates.***

Our last procedure corrected B_{k+1} by using finite differences to try to force $c(B_{k+1}, g_{k+1}) = c(\nabla^2 f(x_{k+1}), g_{k+1})$. By using (3.3), we can instead use finite differences to try to force the stronger condition

$$B_{k+1} g_{k+1} = \nabla^2 f(x_{k+1}) g_{k+1} \, . \tag{3.7}$$

<u>Procedure 2</u>

```
    Given $k \geq 1$, $c_{k-1} > 0$, $x_k$, $x_{k+1}$, $p_k$, $g_k$, $g_{k+1}$, and $B_k$, do the following:
(1) Approximate $\nabla^2 f(x_k)$ with a secant update:
        $y_k = g_{k+1} - g_k$                                        (3.8a)
        $B_{k+1} := U(B_k, p_k, y_k)$ .                               (3.8b)
(2) Estimate the maximum curvature of the problem using (3.1).
(3) If $c(B_{k+1}, g_{k+1}) > c_k$, then apply correction to $B_{k+1}$.
        $\epsilon = \sqrt{macheps}\,(typx)/\,\|g_{k+1}\|$ ,                (3.9a)
        $p_\epsilon = -\,\epsilon\, g_{k+1}$ ,                                (3.9b)
        $y_\epsilon = g(x_{k+1} + p_\epsilon) - g_{k+1}$ .                    (3.9c)
    If $p_\epsilon^T y_\epsilon > 0$, then
        $B_{k+1} := U(B_{k+1}, p_\epsilon, y_\epsilon)$,                      (3.10)
    Else
        $B_{k+1} := (c_k / c(B_{k+1}, g_{k+1}))\, B_{k+1}$ .             (3.11)
    End If.
(4) Exit.
```

A similar procedure has been used by Dixon [9] for constrained optimization with a completely different justification. Let l be the Lagrangian function: $l(x,\lambda)=f(x)+\lambda^T h(x)$, where $h:\mathbb{R}^n \longrightarrow \mathbb{R}^n$ is a set of m equality constraints and $\lambda \in \mathbb{R}^m$ is the set of Lagrange multipliers of the constrained problem. Dixon's method performs an extra update on the approximation to the Hessian of the Lagrangian, denoted $\tilde{B}_k$. This update is of the form

$$\tilde{B}_k = U(\tilde{B}_k, \epsilon\nabla_x l(x_k,\lambda_k),\ \nabla_x l(x_k+\epsilon\nabla_x l(x_k,\lambda_k))-\nabla_x l(x_k,\lambda_k))\,, \tag{3.12}$$

and is included to ensure that the search directions selected are descent directions with respect to a particular merit function.

Both hereditary positive definiteness and the UPD condition are easily established for both of these procedures under very mild conditions on f. The results are as follows.

THEOREM 3. Let $c_0 > 0$ and positive definite B_1 be given. If the update $U(B,p,y)$ used by procedure (1) or (2) satisfies (3.5) and (3.6), then each of these procedures generates a positive definite Hessian approximation at every iteration.

Proof. Since $c_0 > 0$, we have $c_k > 0$ for all k. Now, suppose that for all $i \le k$, B_i is positive definite. Then the "uncorrected" B_{k+1} will be positive definite by hypotheses (3.5) and (3.6). In procedure (1), α is therefore positive and the corrected B_{k+1} will still be positive definite. For the same reason, any corrected B_{k+1} generated by (3.11) will be positive definite, and by hypotheses (3.5) and (3.6), any B_{k+1} generated by (3.10) will be positive definite. Our result follows by induction. □

THEOREM 4. Let $c_0 > 0$, *macheps* > 0, and *typx* > 0 be given. If g is Lipschitz continuous so that for some $L < \infty$, $||g(x)-g(y)|| \le L\,||x-y||$ for all[2] $x, y \in \mathbb{R}^n$, then the curvature estimates $\{c_k\}$ generated by procedures (1) and (2) are uniformly bounded. Furthermore, $\{c(B_k, g_k)\}$ is uniformly bounded above for both procedures.

[2] The assumption of Lipschitz continuity over the whole of $\mathbb{R}^n$ is made for convenience rather than necessity. A smaller region can be used, such as any open convex set that contains the level set of f at x_0.

Proof. From the Cauchy-Schwarz inequality and Lipschitz continuity of g, we have $p_k^T(g_{k+1} - g_k)/p_k^T p_k \leq L$, and hence $c_k \leq \max\{c_0, L\}$. In either procedure, a correction is done if $c(B_{k+1}, g_{k+1}) > c_k$. Now, if $\bar{c}_k \leq 0$ in procedure (1) or $p_\epsilon^T y_\epsilon \leq 0$ in procedure (2), then the correction is simply $B_{k+1} := (c_k / c(B_{k+1}, g_{k+1}))\, B_{k+1}$, resulting in $c(B_{k+1}, g_{k+1}) \leq \max\{c_0, L\}$. The remaining two cases are as follows.

(i) In procedure (1), $\bar{c}_k$ is computed by (3.2b). Now,

$$f_{k+1} - f(x_{k+1} - \epsilon\, g_{k+1}) = \int_0^1 g(x_{k+1} - \lambda\epsilon\, g_{k+1})^T (\epsilon\, g_{k+1})\, d\lambda \tag{3.13}$$

so that

$$\begin{aligned}
\bar{c}_k &= \frac{2}{\epsilon^2 g_{k+1}^T\, g_{k+1}} \int_0^1 (\epsilon\, g_{k+1}^T)(g(x_{k+1} - \lambda\epsilon\, g_{k+1}) - g_{k+1})\, d\lambda \\
&\leq \frac{2}{\epsilon\, \|g_{k+1}\|^2} \int_0^1 \|g_{k+1}\|\ \|g(x_{k+1} - \lambda\epsilon\, g_{k+1}) - g_{k+1}\|\, d\lambda \\
&\leq \frac{2}{\epsilon\, \|g_{k+1}\|} \int_0^1 L\, \|x_{k+1} - \lambda\epsilon\, g_{k+1} - x_{k+1}\|\, d\lambda \\
&\leq L\,.
\end{aligned} \tag{3.14}$$

Hence, if $\bar{c}_k > 0$, B_{k+1} is replaced by $(\bar{c}_k / c(B_{k+1}, g_{k+1}))\, B_{k+1}$, resulting in an approximation which satisfies $c(B_{k+1}, g_{k+1}) \leq L$.

(ii) In procedure (2), a correction is made by an extra secant update provided $y_\epsilon^T\, p_\epsilon > 0$, where p_ϵ and y_ϵ are given by (3.9). The corrected B_{k+1} will satisfy $B_{k+1}\, p_\epsilon = y_\epsilon$. Premultiplying by g_{k+1}^T, using the Cauchy-Schwarz inequality, and invoking the Lipschitz continuity of g allows us to write

$$\begin{aligned}
\epsilon\, g_{k+1}^T B_{k+1} g_{k+1} &= g_{k+1}^T (g(x_{k+1} - \epsilon\, g_{k+1}) - g_{k+1}) \\
&\leq \|g_{k+1}\|^T L\, \|x_{k+1} - \epsilon\, g_{k+1} - x_{k+1}\| \\
&= \epsilon L\, \|g_{k+1}\|^2,
\end{aligned} \tag{3.15}$$

and thus $c(B_{k+1}, g_{k+1}) \leq L$. Therefore, procedures (1) and (2) guarantee that in every situation $c(B_{k+1}, g_{k+1}) \leq \max\{c_0, L\}$. □

The third procedure we present is similar to procedure (1), except any scalings of the Hessian approximations are done *before* the update. This approach is motivated by the *self scaling* algorithms of Oren and Luenberger [17] and the *sizing* algorithm of Dennis, Gay and Welsch [4]. Self scaling algorithms set $\alpha = |p_k^T(g_{k+1} - g_k)|/p_k^T B_k p_k$, or equivalently

$$\alpha = \frac{|p_k^T(g_{k+1} - g_k)|}{p_k^T p_k} \frac{1}{c(B_k, p_k)} . \tag{3.16}$$

The matrix B_{k+1} is then generated by

$$B_{k+1} = U(\alpha B_k, p_k, y_k) . \tag{3.17}$$

Sizing algorithms are similar, except that the modification is only done if $c(B_k, p_k) > |p_k^T(g_{k+1} - g_k)|/p_k^T p_k$. Our procedure is obtained by substituting c_k for $|p_k^T(g_{k+1} - g_k)|/p_k^T p_k$ and $c(B_k, g_{k+1})$ for $c(B_k, p_k)$ in the above formula.

Procedure 3

Given $k \geq 1$, $c_{k-1} > 0$, x_k, x_{k+1}, p_k, g_k, g_{k+1}, and B_k, do the following.

1) Estimate the maximum curvature of the problem using (3.1).

2) Set $\alpha_k = \min\{1, c_k / c(B_k, g_{k+1})\}$.

3) Set $B_{k+1} = U(\alpha_k B_k, p_k, y_k)$ for $y_k = g_{k+1} - g_k$.

4) Exit.

We shall prove results for this procedure only for a specific secant update, the BFGS (see, for example, [7], [8]):

$$U_{BFGS}(B, p, y) = \begin{cases} B & \text{if } y^T p \leq 0 \\ \dfrac{Bpp^T B^T}{p^T Bp} + \dfrac{yy^T}{y^T p} & \textit{otherwise} . \end{cases} \tag{3.18}$$

This is probably the most popular positive definite secant update in use today.

THEOREM 5. Let $c_0 > 0$ and positive definite B_1 be given. Then procedure (3) generates a positive definite Hessian approximation at every iteration if $U \equiv U_{BFGS}$.

Proof. Since $c_0 > 0$, each $c_k > 0$ and every $\alpha_k > 0$. Thus, if B_k is positive definite, αB_k is positive definite. Proof that U_{BFGS} is a positive definite update satisfying (3.5) can be found, for example, in [5]. Since B_1 is positive definite, our result follows by induction. □

The next theorem uses the highly restrictive assumption that f is convex, and is therefore not as strong as our results for procedures (1) and (2). However, this result is stronger than results for the standard BFGS, in that we achieve $c(B_k, g_k) \leq \beta_0$ (which is associated with the strong result $\lim_{k \to \infty} \|g_k\| = 0$) rather than $c(B_k, g_k) \leq \beta_0(1+k)$ (which is associated with the weaker result $\liminf_{k \to \infty} \|g_k\| = 0$.) The proof is almost the same as that used for the standard result (see [19], [21]).

THEOREM 6. Let $c_0 > 0$ and a positive definite matrix B_1 be given. If the Hessian of f is uniformly bounded and f is convex on $\mathbb{R}^n$, then procedure (3) with $U = U_{BFGS}$ generates a sequence of Hessian approximations for which $c(B_k, g_k)$ is uniformly bounded.

Proof. From Theorem 5, we know that each B_k is positive definite. Since $\nabla^2 f(x)$ is uniformly bounded, there exists L such that $\|\nabla^2 f(x)\| \leq L$ for all $x \in \mathbb{R}^n$. Now, $g_{k+1} - g_k = \int_0^1 \nabla^2 f(x_k + \lambda p_k) p_k \, d\lambda$, so we have

$$p_k^T(g_{k+1} - g_k) = \int_0^1 p_k^T \nabla^2 f(x_k + \lambda p_k) p_k \, d\lambda \leq L \, \|p_k\|^2 . \tag{3.19}$$

From this we can deduce that $c_k \leq \max\{c_0, L\}$, and hence $c(\alpha_k B_k, g_{k+1}) \leq \max\{c_0, L\}$.

If $p_k^T y_k \leq 0$, we have $B_{k+1} = \alpha_k B_k$ and $c(B_{k+1}, g_{k+1}) \leq \max\{c_0, L\}$. Otherwise,

$$B_{k+1} = \alpha_k B_k - \alpha_k \frac{B_k p_k p_k^T B_k{}^T}{p_k^T B_k p_k} + \frac{y_k y_k^T}{y_k^T p_k} . \tag{3.20}$$

Now, define $H_k = \int_0^1 \nabla^2 f(x_k + \lambda p_k) d\lambda$ so that $H_k p_k = \int_0^1 \nabla^2 f(x_k + \lambda p_k) p_k \, d\lambda = g_{k+1} - g_k = y_k$.

From the convexity of f, we see that H_k is positive definite or positive semidefinite, and thus there exists $H_k^{½}$ satisfying $(H_k^{½})^T(H_k^{½}) = H_k$. Similarly, there exists $B_k^{½}$ such that $(B_k^{½})^T(B_k^{½}) = B_k$. Substituting these expressions into (3.20) yields

$$B_{k+1} = \alpha_k (B_k^{½})^T \left(I - \frac{(B_k^{½}p_k)(B_k^{½}p_k)^T}{(B_k^{½}p_k)^T(B_k^{½}p_k)} \right) B_k^{½} + (H_k^{½})^T \frac{(H_k^{½}p_k)(H_k^{½}p_k)^T}{(H_k^{½}p_k)^T(H_k^{½}p_k)} H_k^{½} . \quad (3.21)$$

Defining $v_k = B_k^{½}p_k$ and $w_k = H_k^{½}p_k$ leads to

$$\begin{aligned} \hat{g}_{k+1}^T B_{k+1} \hat{g}_{k+1} &= \alpha_k (B_k^{½}\hat{g}_{k+1})^T (I - \hat{v}_k \hat{v}_k^T)(B_k^{½}\hat{g}_{k+1}) + (H_k^{½}\hat{g}_{k+1})^T(\hat{w}_k\hat{w}_k^T)(H_k^{½}\hat{g}_{k+1}) \\ &\le \alpha_k \, ||I - \hat{v}_k\hat{v}_k^T||_2 \, ||B_k^{½}\hat{g}_{k+1}||^2 + ||\hat{w}_k\hat{w}_k^T||_2 \, ||H_k^{½}\hat{g}_{k+1}||^2 \\ &= \alpha_k \, ||B_k^{½}\hat{g}_{k+1}||^2 + ||H_k^{½}\hat{g}_{k+1}||^2 , \end{aligned} \quad (3.22)$$

so that

$$c(B_{k+1}, g_{k+1}) \le c_k + L \le c_0 + 2L \ . \quad \square \quad (3.23)$$

4. Numerical results. The three procedures of the last section were implemented in a trust region algorithm based on the Dennis/Schnabel optimization code [8]. Approximate solutions to the trust region subproblem (2.3) were computed with an optimal locally constrained procedure ([8], [10], [12], [16]). The BFGS method was used to generate updates $U(B, p, y)$.

Twenty-six problems were selected from the standard test set of Moré, Garbow, and Hillstrom [14]. These problems were *NPROB* = 1, 2, ..., 18 with the default problem dimensions, and the variable dimension problems 6, 7, 8, 9, 13, 14, and 15 with dimensions n = 10, 9, 18, 6, 6, 10 and 20 respectively. An "easy quadratic" problem with n = 4 was included as an additional control.

Table (1) shows a representative sample of the results obtained by our procedures when the algorithm was executed from the standard starting point. The numbers 0 to 3 in the first column represent, respectively, the unmodified BFGS, procedure (1), procedure (2), and procedure (3). The next two columns represent the problem number and dimension. "Itn" represents the number of iterations performed, and "C" represents the number of corrections done. The "#f" and "#g" columns represent the number of function and gradient evaluations required. The "$f*$" and "$||g*||$" columns represent the function value and the norm of the gradient at the final iteration.

The two numbers in the "performance measure" column represent two different ways of measuring algorithm performance. Criteria (A) is defined as the sum of the number of function and gradient evaluations, while criteria (B) is the number of function evaluations plus n times the number of gradient evaluations. Thus the first column is the best measure if gradient evaluations are inexpensive, while the second column is the best measure if gradient evaluations are much more expensive than function evaluations.

The results for all twenty-six test problems are summarized in Table (2). Corrections were made in 10-20% of the iterations[3]. Even though these corrections required an extra function evaluation in procedure (1) and an extra gradient evaluation in procedure (2), the reduction in total number of iterations is sufficiently large to actually reduce the number of equivalent function evaluations. Problem 8 with $n = 18$ is the most striking example of this. The algorithm using the unmodified BFGS is still far from the solution after 200 iterations. Procedures (1) and (3) have not converged either, but are much closer (as measured by $||g_{200}||$ and f_{200} being roughly 100 times smaller for procedures (1) and (3) than for the BFGS). Procedure (2) is the clear winner, having converged in 89 iterations.

Table (3) presents the data from Table (2) in a normalized format. Each of the performance indicators shown in the table is normalized by the value from the unmodified BFGS method with the exception of the second column under "normalized count of corrections," which instead denotes the fraction of iterations at which a correction was performed. The first 4 rows represent the complete data set, while rows 5 to 8 represent only those test cases for which *all* of the methods succeeded.[4] We see that options(1) and (3) improved most performance indicators by roughly 10%, while procedure (2) increased performance by 15 to 20%.

To investigate how reliable our procedures were, we ran the previous set of test cases again with different starting points (10* and 100* the default values) and included an additional 12

[3] It is worth mentioning that (3.11) (the naive correction done in procedure (2) if the secant correction cannot be guaranteed to produce a positive definite matrix) was never invoked in any of our test runs.

[4]It can be argued that this reduced data set gives a better measure of the relative efficiency of algorithms.

Table (1)

Performance of algorithms on representative problems.

P	Pb	n	Itn	C	$\# f$	$\# g$	Performance measure (A)	Performance measure (B)	$f(x^*)$	$\|\|g(x^*)\|\|$
0	1	3	27	0	36	27	63	117	0.14d-18	0.68d-08
1			25	2	38	25	63	113	0.47d-15	0.13d-06
2			26	4	41	30	71	131	0.30d-14	0.10d-05
3			27	8	37	27	64	118	0.36d-14	0.19d-05
0	2	6	45	0	49	45	94	319	0.57d-02	0.27d-05
1			35	3	45	35	80	255	0.57d-02	0.99d-06
2			35	3	41	38	79	269	0.57d-02	0.21d-05
3			36	4	47	36	83	263	0.57d-02	0.24d-05
0	5	3	29	0	39	29	68	126	0.19d-18	0.15d-08
1			19	13	39	19	58	96	0.62d-19	0.87d-10
2			20	6	27	26	53	105	0.86d-16	0.34d-07
3			18	7	21	18	39	75	0.11d-13	0.15d-06
0	8	2	104	0	137	104	241	345	0.84d-05	0.40d-05
1			77	1	99	77	176	253	0.84d-05	0.38d-05
2			42	1	64	43	107	150	0.84d-05	0.89d-06
3			77	1	118	77	195	272	0.84d-05	0.14d-05
0		18	200*	0	203	200	403	3803	0.47d-01	0.59d+00
1			200*	2	231	200	431	3831	0.17d-03	0.21d-03
2			89	2	122	91	213	1760	0.14d-03	0.14d-05
3			200*	36	210	200	410	3810	0.17d-03	0.75d-03
0	13	2	12	0	13	12	25	37	0.20d-13	0.26d-06
1			11	2	14	11	25	36	0.12d-10	0.47d-05
2			9	3	11	12	23	35	0.26d-14	0.11d-06
3			11	1	12	11	23	34	0.19d-12	0.59d-06
0		6	21	0	23	21	44	149	0.27d-03	0.87d-05
1			28	4	37	28	65	205	0.27d-03	0.17d-05
2			17	12	19	29	48	193	0.27d-03	0.11d-05
3			24	6	31	24	55	175	0.27d-03	0.17d-05
0	15	4	48	0	54	48	102	246	0.61d-09	0.14d-05
1			27	6	37	27	64	145	0.12d-08	0.58d-05
2			33	6	37	39	76	193	0.31d-08	0.50d-05
3			32	4	38	32	70	166	0.14d-09	0.50d-06
0		20	48	0	54	48	102	1014	0.30d-08	0.31d-05
1			33	11	48	33	81	708	0.16d-10	0.85d-05
2			33	6	37	39	76	817	0.15d-07	0.11d-04
3			38	10	51	38	89	811	0.16d-09	0.41d-05
0	17	4	103	0	148	103	251	560	0.39d-14	0.25d-05
1			31	3	54	31	85	178	0.21d-16	0.12d-06
2			31	4	42	35	77	182	0.59d-14	0.16d-05
3			86	9	120	86	206	464	0.16d-15	0.27d-06

* Algorithm exceeded maximum number of iterations allowed.

Table (2)
Summary of performance of algorithms on 26 test problems.

Procedure	Number of iterations	Number of corrections	Number of evaluations f	g	Performance measures (A)	(B)
0	1047	0	1309	1047	2356	9525
1	902	108	1329	902	2231	8963
2	696	111	946	807	1753	6637
3	939	186	1268	939	2207	8923

Table (3)
Test data normalized with respect to control (unmodified BFGS).

Procedure	Normalized count of iterations	corrections	Normalized count of evaluations f	g	Normalized performance measures (A)	(B)
0	1.00*	0.00	1.00	1.00	1.00	1.00
1	.86*	.12	1.00	.86	.95	.94
2	.66*	.16	.72	.77	.74	.70
3	.90*	.20	.97	.90	.94	.94
0	1.00+	0.00	1.00	1.00	1.00	1.00
1	.83+	.14	.98	.83	.91	.89
2	.72+	.17	.75	.84	.79	.85
3	.86+	.18	.93	.86	.90	.88

* Data from all 26 test problems.

+ Data from reduced set of the 23 test problems for which all methods converged.

problems of larger dimension ($n = 10$ to $n = 68$) . The unmodified BFGS failed[5] on 21 out of the 144 cases while procedures (1), (2), and (3) failed on 16, 12, and 20 respectively. Thus, at least for this test set, procedure (1) increases the reliability of the BFGS method by 25%, and procedure (2) increases the reliability by 45%. Other performance indicators for this expanded test set are roughly comparable with those shown in Table (3), with the exception that procedures (1) and (3) performed relatively poorly on the large dimensional problems.[6]

Given the excellent numerical performance of procedure (2), the remainder of our investigations focus on this technique. In order to examine performance of this procedure when invoked with a greater or lesser frequency, we devised alternative tests for triggering the extra update. The first such test defines

[5] Several criteria were used to define "failure", such as still being far from the solution after 200 iterations.

[6] Each option failed on the same problems that the unmodified BFGS failed on. Furthermore, procedure (1) was less efficient by a factor of 30%. However, this is not conclusive due to the small size of the problem set.

$$c_k = \max\{m_2 c_{k-1}, p_k^T(g_{k+1} - g_k)/p_k^T p_k\} \tag{4.1}$$

and triggers an extra update whenever

$$c(B_{k+1}, g_{k+1}) > m_1 c_k, \tag{4.2}$$

where $m_1 \in [0, \infty]$ and $m_2 \in [0, 1]$ are constants. The constant m_2 controls how much "memory" the test has of previous curvature estimates. For example, $m_2 = 0$ corresponds to $c_k = \max\{0, p_k^T(g_{k+1} - g_k)/p_k^T p_k\}$: the current curvature estimate. The constant m_1 allows a more direct control of how often corrections are triggered: $m_1 = \infty$ corresponds to the unmodified BFGS method, while $m_1 = 0$ corresponds to performing an extra update at every iteration. We tested this new triggering formula on the original 26 problems using two different starting points for each (1 * and 10 * the default values) with the values $m_1 \in \{\infty, 4, 2, 1, 0.5, 0.25, 0\}$ and $m_2 \in \{1, 0.9, 0.7, 0.5, 0.01\}$.

Several trends were noticed among the performance indicators as m_1 was varied from ∞ to 0. The percentage of iterations at which the model Hessian B_k was corrected varied monotonically from 0% to 95%,[7] but the number of iterations required (as compared to the unmodified BFGS) varied monotonically from 100% to 55%. The performance indicators for $m_1 \geq 2$ showed little change from the BFGS in efficiency.[8] For $1 \geq m_1 \geq 0.25$, the efficiency was improved by roughly 15% (using criteria (A)) and by 10% (using criteria (B)); for $m_1 = 0$, the efficiency was unchanged (using criteria[9] (B)) and improved by 20% (using criteria (A)). The number of failures dropped from 8 to 2. Experiments with m_2 showed similar (although slightly less clear) trends.

These trends can be summarized by saying the algorithm performs quite well for $m_1 \in [0, 1]$. A value of $m_1 = ½$ and $m_2 = 1$ is suggested for typical problems, although $m_1 \in [0, ½]$ and $m_2 = 1/100$ may be preferable if the gradient is very inexpensive to evaluate or if reliability is very important. Table (4) presents a small sample of our data from this part of the investigation.

[7]Corrections were not made at the initial and final iterations.

[8]B_k was modified in fewer than 5% of the iterations, which is apparently not enough to improve efficiency. However, even this small percentage was enough to roughly halve the failure rate for this test set.

[9]This is hardly surprising, since we have roughly halved the number of iterations while doubling the number of gradient evaluations per iteration.

Table (4)
Test of triggering corrections using equations (4.1) and (4.2).

m_1	m_2	Normalized count of iterations	Normalized count of corrections	Normalized count of evaluations f	Normalized count of evaluations g	Normalized performance measures (A)	Normalized performance measures (B)	Normalized count of failures
∞	-	1.00*	0.00	1.00	1.00	1.00	1.00	1.00
1.0	1.00	0.75*	0.09	0.76	0.82	0.79	0.79	0.50
0.5	1.00	0.79*	0.18	0.84	0.93	0.88	0.86	0.38
0.5	0.01	0.55*	0.95	0.57	1.07	0.79	1.03	0.25
∞	-	1.00+	0.00	1.00	1.00	1.00	1.00	-
1.0	1.00	0.83+	0.11	0.83	0.92	0.87	0.90	-
0.5	1.00	0.76+	0.24	0.80	0.95	0.87	0.90	-
0.5	0.01	0.56+	0.94	0.55	1.08	0.78	1.02	-

* Data from full set of 52 test cases.
+ Data from reduced set of the 43 test problems for which all methods converged.

An alternative to the test $c(B_{k+1}, g_{k+1}) > c_k$ is suggested by equations (2.15) and (2.16) in Theorem 2. From the secant equation and the Lipschitz continuity of g, we have that the unmodified update satisfies

$$p_k^T B_{k+1} p_k \leq L \, \|p_k\|^2 \tag{4.3}$$

so that

$$c(B_{k+1}, p_k) \leq L \ . \tag{4.4}$$

Let $\Theta_{\max}$ be any fixed angle in $(0, \frac{\pi}{2})$, and let Θ_{k+1} be the angle between p_k and g_{k+1}. If $\cos\Theta_{k+1} \geq \cos\Theta_{\max}$ and if $p_k \in S_{k+1}$, then by (2.16) we have the uniform predicted decrease condition (2.3) at the next iteration without requiring a correction (β_0 and β_1 are taken to be $L/\cos\Theta_{\max}$ and $\beta_2 \cos\Theta_{\max}$, respectively). For example, if $\Theta_{k+1} \geq \Theta_{\max} = \frac{\pi}{4}$ and $\beta_2 = 1$, we have that

$$\begin{aligned} pred_{k+1}(p_{k+1}) &\geq \max\{pred_{k+1}(p) \colon p = \alpha p_k, \ \|p\| \leq \Delta_{k+1}\} \\ &\geq \frac{1}{2}\cos\Theta_{k+1} \|g_{k+1}\| \min\left\{\Delta_{k+1}, \cos\Theta_{k+1}\frac{\|g_{k+1}\|}{L}\right\} \\ &\geq \frac{1}{2}(\frac{\sqrt{2}}{2}) \|g_{k+1}\| \min\left\{\Delta_{k+1}, \frac{\|g_{k+1}\|}{\sqrt{2}L}\right\}. \end{aligned} \tag{4.5}$$

For OLC type methods of computing trial steps, S_k is taken to be $\mathbb{R}^n$; hence for these methods we can trigger a correction if

$$\cos \Theta_{k+1} < \cos \Theta_{\max} . \tag{4.6}$$

Numerical tests were performed using this alternative criteria (by itself and in several Boolean combinations with the original test) for various values of $\Theta_{\max}$. Although this new trigger was also successful in improving performance and reliability, our results were somewhat erratic as $\Theta_{\max}$ was changed in comparison to the uniform performance increases exhibited by the original trigger as m_1 and m_2 were changed (within the interval (0, 1]). At present, we therefore recommend the original test over this alternative.

5. Concluding remarks.

5.1. Summary. An important element in the global convergence theory of trust region algorithms is the uniform predicted decrease condition, which is usually established by assuming a uniform upper bound on the sequence of Hessian approximations. However, uniform predicted decrease is also implied by the weaker condition of a uniform upper bound on $c(B_k, g_k)$. This condition can be enforced in secant methods by several procedures. The best one of these procedures involves performing an extra secant update to correct B_k in the gradient direction at every iteration at which $c(B_k, g_k)$ is larger than an estimate of the maximum problem curvature. Numerical testing suggests that this technique increases the reliability of secant methods by a factor of two or more, while actually increasing several measures of algorithm efficiency by 10 to 20 percent.

It should be pointed out that this increase in efficiency is a surprising result, particularly since the increase is remarkably insensitive to how frequently corrections are performed. Our initial goal was merely to find a procedure which increased the reliability of secant methods without immoderately *decreasing* the efficiency. We also expected our procedure to work best when corrections were done rarely; our experiments showed instead the utility of frequent extra updates.

Another point that should be restated is that the uniform predicted decrease condition is not, of itself, sufficient to imply global convergence. This condition must be coupled with conditions about how the trust radius is updated at each iteration. Specifically, global convergence can be shown if (2.3) holds and there exists $\epsilon_0 > 0$ such that

$$\Delta_k \leq \epsilon_0 \|g_k\|/\beta_0 \Rightarrow \Delta_{k+1} \geq \Delta_k \; . \tag{5.1}$$

Traditional global convergence proofs establish (5.1) by again using the assumption of a uniform upper bound on the sequence of Hessian approximations, so that the weaker condition (of bounded Rayleigh quotients in the gradient directions) used in this paper might seem irrelevant with respect to the overall theory. However, (5.1) can also be established (see [2]) by a weaker condition: that (2.3) holds and that there exists $\beta_3 \in [0,\infty)$ such that

$$-\beta_3 \leq c(B_k, p) \quad \textit{for all nonzero} \quad p \in \mathbf{R}^n \; . \tag{5.2}$$

That is, global convergence can be established if $c(B_k, g_k)$ is bounded above and if the eigenvalues of B_k are bounded away from $-\infty$. For positive definite secant methods, (5.2) is automatic and the procedures of this paper guarantee global convergence.

5.2. Extensions. The details of the procedures we have presented have been purposefully kept simple for clarity; many refinements are possible. For example, the calculation of the finite difference steplength variable ϵ was dependent on a fixed parameter *typx*, which represented an estimate of the typical size of $\|x_k\|$. A better approach would be to prespecify some *typmin* > 0 and use $typx_k = \max\{\|x_k\|, \frac{1}{2}(\|x_k\| + \|x_{k-1}\|), typmin\}$. A variety of other strategies for computing ϵ are discussed [8] and [15].

A more important topic not yet mentioned is the subject of scaling matrices. Most implementations of trust region methods replace the spherical trust region heretofore assumed with the hyperellipse $\|D_k p\| \leq \Delta_k$ where D_k is a nonsingular scaling matrix. The trust region subproblem is then

$$\underset{p \in \mathbf{R}^n}{\textit{minimize}} \; \psi_k(x_k + p): \; \|D_k p\| \leq \Delta_k \; . \tag{5.3}$$

This can be converged back to standard form by making the change of variables

$$\tilde{x} = D_k x \tag{5.4}$$

so that $\tilde{x}_k = D_k x_k$, $\tilde{p} = D_k p$, etc. Defining

$$\tilde{\psi}_k(\tilde{x}_k + \tilde{p}) \equiv \psi_k(x_k + p) = f_k + \tilde{g}_k^T \tilde{p} + \frac{1}{2}\tilde{p}^T \tilde{B}_k \tilde{p} \tag{5.5}$$

with

$$\tilde{g}_k = D_k^{-T} g_k \tag{5.6}$$

and

$$\tilde{B}_k = D_k^{-T} B_k D_k^{-1} \tag{5.7}$$

allows us to replace (5.3) with the problem

$$\underset{\tilde{p} \in \mathbb{R}^n}{minimize}\ \tilde{\psi}_k(\tilde{x}_k + \tilde{p}) \colon \ \|\tilde{p}\| \le \Delta_k \ . \tag{5.8}$$

A step p_k can then be obtained by computing an approximate solution $\tilde{p}_k$ to problem (5.8) and inverting transformation (5.4). If $\{D_k\}$ and $\{D_k^{-1}\}$ are uniformly bounded, the global convergence results of this paper can still be obtained provided there exists $\beta_0 < \infty$ such that

$$c(\tilde{B}_k, \tilde{g}_k) \le \beta_0 . \tag{5.9}$$

Our safeguarding techniques should therefore be used to enforce either

$$c(\tilde{B}_k, \tilde{g}_k) \le \tilde{c}_k \tag{5.10a}$$

with $\tilde{c}_k = \max\{\, \tilde{c}_{k-1} \, , \tilde{p}_k^T(\, \tilde{g}_{k+1} - \tilde{g}_k\,) \,/\, \tilde{p}_k^T \tilde{p}_k \,\}$, or

$$c(\, B_k \, , (D_k^T D_k)^{-1}\, g_k \,) \le c_k \ , \tag{5.10b}$$

depending on which is more convenient for a given implementation.

Some other possible refinements are based on secant updates which preserve past information. Davidon [3] and Schnabel [23] have developed classes of updates which satisfy

$$B_{k+1}\, p_k = g_{k+1} - g_k \quad \text{and} \quad (B_{k+1} - B_k) U_k = 0 \, , \tag{5.11}$$

where $U_k \in \mathbb{R}^{n \times m}$ is a matrix whose columns form an orthonormal basis for the space spanned by one or more previous search directions. Sorensen [25] suggests a broader class of updates which satisfy the weaker conditions

$$B_{k+1}\, p_k = g_{k+1} - g_k \quad \text{and} \quad U_k^T (B_{k+1} - B_k) U_k = 0 \ . \tag{5.12}$$

One application of an update which can satisfy (5.11) is obvious: under many conditions our extra update in procedure (2) can be executed so that

$$B_{k+1}\,p_\epsilon = y_\epsilon \quad \text{and} \quad (B_{k+1} - \tilde{B}_{k+1})p_k = 0 \tag{5.13}$$

(where $\tilde{B}_{k+1} = U(B_k, p_k, y_k)$), and hence our new approximation to the Hessian satisfies both $B_{k+1}\,p_k = y_k$ and $B_{k+1}\,p_\epsilon = y_\epsilon$. Similarly, procedure (3) can be modified using updates satisfying either (5.11) or (5.12) so that our Hessian approximation satisfies both $B_{k+1}\,p_k = y_k$ and $g_{k+1}^T\, B_{k+1} g_{k+1} / g_{k+1}^T\, g_{k+1} = \bar{c}_k$ (where $\bar{c}_k$ is computed as in procedure (1)).

If updates satisfying (5.11) or (5.12) are used, the alternative trigger (4.6) previously suggested might also become competitive with triggering extra updates whenever $c(B_{k+1}, g_{k+1}) \geq c_k$. Rather than defining Θ_{k+1} to be the angle between g_{k+1} and p_k, we can define it to be the angle between g_{k+1} and the "nearest" element of *span* $\{p_k, p_{k-1}, \ldots, p_{k-m}\}$. Specifically, $\cos\Theta_{k+1} = g_k^T(Q_k\, g_k)/\,\|g_k\|\,\|Q_k\, g_k\|$ where Q_k is the projection matrix $U_k\, U_k^T$.

This list of possible refinements to our techniques is not meant to be exhaustive, but should be indicative of topics for future research. Our numerical tests were confined to the BFGS method, but our procedures can be applied to other secant methods, such as the DFP update and the "rank one" update. Safeguarding seems particularly appropriate for these two updates, since they are *usually* very effective, but under certain circumstances may generate Hessian approximations with inappropriately large eigenvalues. A numerical comparison of the BFGS, DFP, and rank one updates with and without safeguarding by procedure (2) is currently being performed. Our procedures can also be easily extended to such classes of updates as sparse secant methods and structured secant methods. A description of procedure (2) as applied to a structured secant method associated with nonlinear least squares problems can be found in [1].

[10]Depending on how the algorithm is defined, this set may also include p_ϵ steps from one or more past iterations.

REFERENCES

(1) CARTER, R.G. [1986]. Multi-model algorithms for optimization, TR86-3, Department of Mathematical Sciences, Rice University, Houston, Texas.

(2) CARTER, R.G. [1987]. Global convergence theory for linesearch and trust region methods, TR87-16, Department of Mathematical Sciences, Rice University, Houston, Texas.

(3) DAVIDON, W.C. [1975]. Optimally conditioned optimization algorithms without exact line searches, *Math. Prog.*, 9, pp. 1-30.

(4) DENNIS, J.E. Jr., D.M. GAY and R.E. WELSCH [1981a]. An adaptive nonlinear least-squares algorithm, *TOMS*, 7, pp. 348-368.

(5) DENNIS, J.E. Jr., D.M. GAY and R.E. WELSCH [1981b]. Algorithm 573 NL2SOL - An adaptive nonlinear least-squares algorithm [E4], *TOMS*, 7, pp. 369-383.

(6) DENNIS, J.E. Jr. and H.H. MEI [1979]. Two new unconstrained optimization algorithms which use function and gradient values, *J. Optim. Theory Appl.*, 28, pp. 453-482.

(7) DENNIS, J.E. Jr. and R.B. SCHNABEL [1979]. Least change secant updates for quasi-Newton methods, *SIAM Review*, 21, pp. 443-459.

(8) DENNIS, J.E. Jr. and R.B. SCHNABEL [1983]. *Numerical Methods for Unconstrained Optimization and Nonlinear Equations*, Prentice-Hall, Englewood Cliffs, New Jersey.

(9) DIXON, L.C.W. [1979]. Exact penalty function methods in nonlinear programming, TR 103, Hatfield Polytechnic Optimization Centre.

(10) GAY, D.M. [1981]. Computing optimal locally constrained steps, *SIAM J. Sci. Statist. Comput.*, 2, pp. 186-197.

(11) GOLDFELDT, S.M., R.E. QUANDT, and H.F. TROTTER [1966]. Maximization by quadratic hill-climbing, *Econometrica*, 34, pp. 541-551.

(12) HEBDEN, M.D. [1973]. An algorithm for minimization using exact second derivatives, Technical report TP515, Atomic Energy Research Establishment, Harwell, England.

(13) MORÉ, J.J. [1982]. Recent developments in algorithm and software for trust region methods, Argonne National Labs Report ANL/MCS-TM-2.

(14) MORÉ, J.J., B.S. GARBOW and K.E. HILLSTROM [1981]. Testing unconstrained optimization software, *TOMS*, 7, pp.17-41.

(15) MORÉ, J.J. and D.C. SORENSEN [1982]. Newton's method, Argonne National Labs Report ANL-82-8, Argonne, Illinois.

(16) MORÉ, J.J. and D.C. SORENSEN [1983]. Computing a trust region step, *SIAM J. Sci. Statist. Comput.*, 4, pp. 553-572.

(17) OREN, S.S. and D.G. LUENBERGER [1974]. Self-scaling variable metric (SSVM) algorithms. I. Criteria and sufficient conditions for scaling a class of algorithms, *Manage. Sci.* 20, pp. 845-862.

(18) POWELL, M.J.D. [1970]. A new algorithm for unconstrained optimization, in *Nonlinear Programming,* J.B. Rosen, O.L. Mangasarian and K. Ritter, eds., Academic Press, New York, pp. 31-65.

(19) POWELL, M.J.D. [1972]. Some properties of the variable metric algorithm, in *Numerical methods for nonlinear optimization,* F.A. Lootsma, ed., Academic Press, London.

(20) POWELL, M.J.D. [1975]. Convergence properties of a class of minimization algorithms, in *Nonlinear Programming 2* O.L. Mangasarian, R.R. Meyer, S.M. Robinson, eds., Academic Press, New York, pp. 1-27.

(21) POWELL, M.J.D. [1976]. Some global convergence properties of a variable metric algorithm without exact line searches, in *Nonlinear Programming,* R. Cottle and C. Lemke, eds., AMS, Providence, Rhode Island, pp. 53-72.

(22) POWELL, M.J.D. [1984]. On the global convergence of trust region algorithms for unconstrained minimization, *Math. Programming,* 29, pp. 297-303.

(23) SCHNABEL, R.B. [1977]. Analysing and improving quasi-Newton methods for unconstrained optimization, Ph.D. thesis, Cornell Computer Science TR-77-320.

(24) SCHULTZ, G.A., R.B. SCHNABEL and R.H. BYRD [1985]. A family of trust-region-based algorithms for unconstrained minimization with strong global convergence properties, *SIAM J. Numer. Anal.,* 22, pp. 47-67.

(25) SORENSEN, D. [1980]. The Q-superlinear convergence of a collinear scaling algorithm for unconstrained optimization, *SIAM J. Numer. Anal.,* 17-1, pp. 84-114.

(26) SORENSEN, D.C. [1982a]. Trust region methods for unconstrained minimization, in *Nonlinear Optimization* 1981, M.J.D. Powell, ed., Academic Press, London.

(27) SORENSEN, D.C. [1982b]. Newton's method with a model trust region modification, *SIAM J. Numer. Anal.,* 19, pp. 409-426.

(28) STEIHAUG, T. [1981]. The conjugate gradient method and trust regions in large scale optimization, Department of Mathematical Sciences, TR81-1, Rice University.